UVM Based Design and Verification of AHB to APB Protocol

Venkatesh V[1], Vipul Pyati[2], Vishal Patil[3], Y. Sai Rahul[4], Dr. Kumar P[5]
[1,2,3,4] *Student, Department of Electronics and Communication Engineering, DSCE, Bengaluru, India*
[5]*Assistant Professor, Department of Electronics and Communication Engineering, DSCE, Bengaluru, India*

Abstract **- The project focuses on the design and verification of a bridge protocol that connects the AHB (Advanced High- performance Bus) to the APB (Advanced Peripheral Bus) using UVM (Universal Verification Methodology). It aims to create a reliable interface between these two commonly used bus architectures. The design process includes implementing the bridge log ic in UVM, addressing critical elements such as data transfer, address mapping, and control signals. Various verification techniques, such as simulation and testing, are employed to ensure the bridge protocol's accuracy and reliability. The AHB to APB bridge plays a crucial role in integrating different bus protocols and enabling efficient communication between high- performance processing units and slower peripherals, contributing to the overall effectiveness and functionality of the SoC. This work focuses on functional verification of AMBA AHB to APB Bridge protocol for completeness by employing System Verilog layered testbench architecture. This ensures complete verification of functionality with maximal coverage.**

Keywords: AMBA, VLSI, VIP, SoC, APB, UVM, Design Verification.

1. INTRODUCTION

With the advancement of deep-submicrometric technology, it is now possible to design and build a system-on-a-chip (SoC) with several intellectual-property (IP) cores while fulfilling short time-to-market requirements.

The process of designing and verifying the AHB to APB bridge protocol with the Universal Verification Methodology (UVM) involves developing a hardware description of the bridge and ensuring compliance with the required communication standards and functionalities. UVM is a standardized approach used for verifying integrated circuit designs.

The AMBA AHB is designed for high-performance and high clock frequency system modules, serving as the main system backbone bus.

It supports the efficient integration of processors, on-chip memories, and off-chip external memory interfaces with low-power peripheral microcell functions. Additionally, AHB is designed to streamline the design process, facilitating synthesis and automated testing techniques.

The AMBA ASB (Advanced System Bus) serves as an alternative to the AHB for scenarios where high-performance features are unnecessary. Similar to AHB, the ASB supports the efficient integration of processors, on- chip memories, and off- chip external memory interfaces, along with low-power peripheral macro cells.

Designing and verifying an AHB to APB bridge protocol using the Universal Verification Methodology (UVM) is crucial for facilitating effective communication between high- performance processors and peripheral devices within a System-on-Chip (SoC).

SoC designs typically involve the integration of components from various Intellectual Property (IP) providers. An AHB to APB Bridge offers a standardized interface that eases the integration of diverse IP blocks

The goal of functional verification is to make sure that the design operates as intended. Coverage is employed to evaluate functional verification. Functional verification has two types of coverage: functional coverage and code coverage. The quantity of code that is turned on during the verification process is referred to as code coverage. It may help to identify false pathways and dead code that doesn't

seem active on particular inputs. Functional coverage shows the number of successful transactions

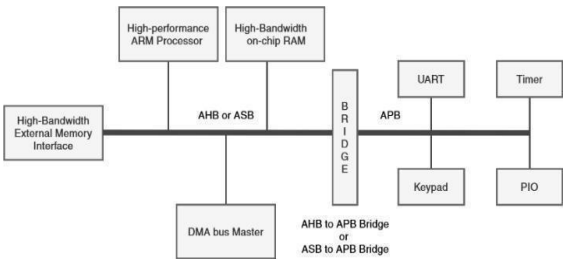

Figure 1: AMBA Bus Architecture

Figure 1 shows the AMBA architecture. Notice the presence of two buses, namely AHB and APB buses. As the CPU(ARM) cores, DMA, high bandwidth memory demand high performance, they are connected to AHB bus while the low bandwidth peripherals are interconnected through APB bus [7]. There is an AHB to APB bridge which connects AHB and APB. All APB connected peripherals act as slaves while the AHB-APB bridge (simply APB bridge) acts as the Master and initiates all the transactions [8].

2. METHODOLOGY

The methodology for the design and verification of the AHB to APB bridge protocol using UVM involves a structured approach that ensures thorough testing and high design reliability. The process begins with the creation of a layered UVM testbench, including key components such as agents, drivers, sequencers, monitors, and a scoreboard. Each component is tailored to handle specific tasks: agents facilitate communication between the testbench and the DUT (Design Under Test), drivers convert stimulus into protocol- specific signals, and monitors log the DUT's responses. Sequencers generate a variety of test cases, including both randomized and directed scenarios, to validate different transaction types like single and burst read/write operations

We are using a UVM-based approach to create a modular and reusable testbench for verifying the AHB to APB Bridge. This approach allows for efficient debugging, coverage analysis, and scalability

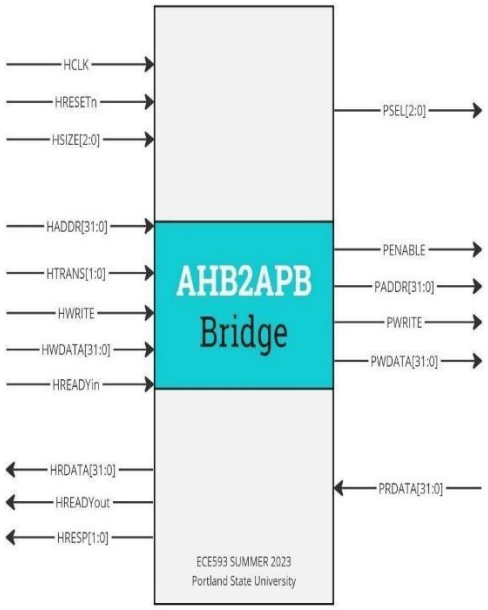

Figure 2: AHB to APB Bridge

2.1 AHB

AMBA bus protocols are essential for SoCs and ASICs because they connect and manage functional blocks. AHB Lite is a subset of a larger interconnect standard named AHB. AHB supports a single master and multiple slave modules for high-frequency and high-performance systems.

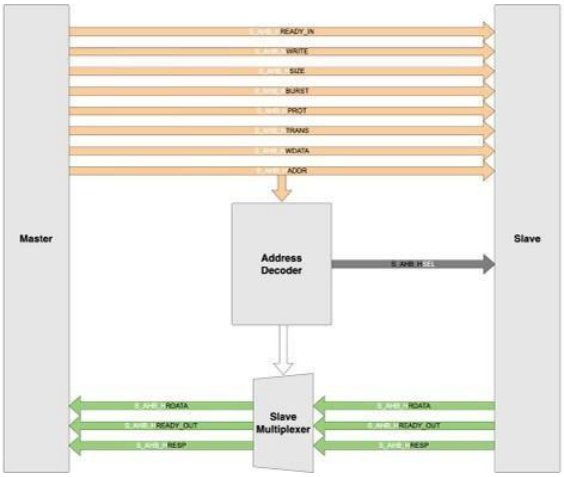

Figure 3: AHB

The AHB system consisting of a master, slave, decoder, and multiplexer. To begin a read or write operation, the master provides data that contains the target slave address, transfer width, transfer type, and direction. The decoder translates the address and enables the appropriate slave peripheral. A decoder is necessary for any AHB-Lite system with more than one slave. When the slave receives this APB Slave data, the slave sends a signal back to the master

through the multiplexer, stating whether the transfer was a success, failure, or is still in progress. An additional output signal from the decoder connects to the multiplexer's select signal [5]. This lets the multiplexer know which slave signal to send to the master. Similarly, a multiplexer is necessary for any AHB Lite system with multiple slaves.

2.2. APB

APB Slave Description The APB Slave has a simple yet flexible interface, making it suitable for connecting multiple slaves and subsequent functions. On the positive edge of the clock, when both the select signal and enable signal are high, the address and write signal can determine which register should be updated by the write operation.For read transactions, data can be driven onto the data bus when the write signal is low, and both the select and enable signals are high.

OPERATING STATUS

IDLE
This is the default state where no data transfer occurs.

SETUP
In this state, a suitable Select signal is asserted. The bus remains in the SETUP state for only one clock cycle and then transitions to the ENABLE state on the next positive edge of the clock.
Enable/Access

- When the select input and write signal are high, and the ready signal is also high, the Master intends to write data to the slave devices. After the transaction, the ready signal goes low, indicating that the slave is not ready for further transfers.
- If the ready signal is high but the select signal is low and the write signal is low, no transfer takes place, and the bus remains in a wait state

3. VERIFICATION USING UVM

The Universal Verification Methodology (UVM) is a robust framework used for designing and verifying complex digital systems. It offers substantial advantages through reusable and scalable test benches. UVM facilitates reusability by providing a standardized approach for creating modular and configurable verification components.

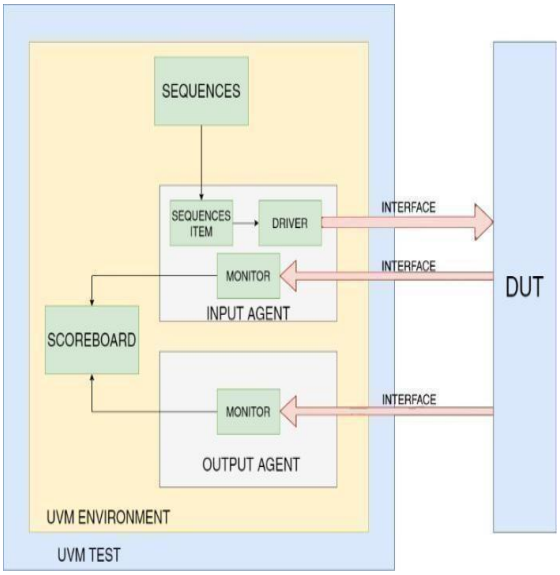

Figure 4: UVM Environment

UVM Agent
Agents serve as the interface between the test bench and the Design Under Test (DUT). They facilitate stimulus generation and response capture. Typically, agents are structured into separate components such as sequencers, drivers, and monitors

Sequencer
Sequencers generate sequences of transactions or stimuli that are sent to the DUT. They control the flow of data and synchronization, managing the sequencing of transactions based on defined test scenarios in the testbench.

Monitor
Monitors observe and capture signals or responses from the DUT's interface during simulation. They provide visibility into the behavior and operation of the DUT, capturing data used for analysis and comparison with expected results.

Environment
Environments manage the hierarchy of components in the testbench, including agents, sequencers, monitors, and scoreboards. They oversee the overall verification flow, handle inter-component communication, and orchestrate the execution of test scenarios.

Sequence Item
Sequence items represent transactions or packets of data exchanged between the testbench and the DUT. They encapsulate stimulus information and may

include fields and methods for configuring and manipulating data during simulation.

Cover Group

A cover group is a user-defined construct used for coverage collection in UVM. It defines a type that can be instantiated multiple times in different contexts. Similar to a class, once defined, a cover group instance can be created using the `new ()` operator. Cover groups help in tracking and reporting on the completeness of verification by monitoring specified aspects of the DUT's behavior.

4. RESULT AND DISCUSSION

The AMBA AHB to APB Bridge protocol's behavior is examined in a System Verilog verification environment. The verification tests' objective is to simulate the protocol's transfer like read and write. The address is decoded and driven onto PADDR during this state, the associated PSEL line is driven HIGH, and PWRITE is driven HIGH.

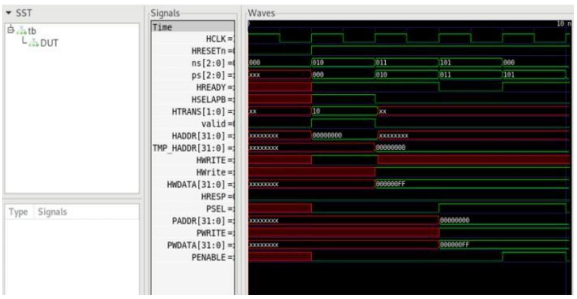

Figure. 5. Write transfer

The address is decoded and driven onto PADDR during this state, so the appropriate PSEL line is driven HIGH, and PWRITE is driven LOW. A wait state is constantly included to ensure that the current AHB transfer's data phase does not end until the APB read data has been driven onto HRDATA.

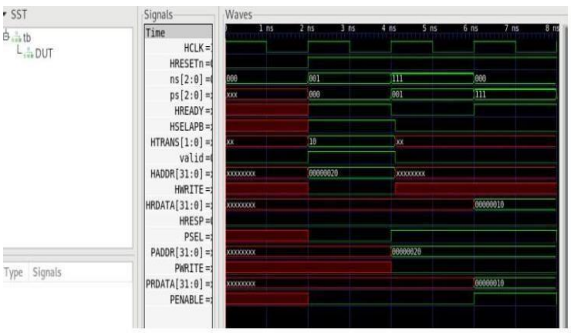

Figure 6: Read Transfer

In incrementing bursts, the address of each transfer in the burst is an increment of the preceding address. The length of the bursts is determined by HSIZE. HBRUST shows whether the burst is single or incrementing.

Figure 7: Burst Read Transfer

5. CONCLUSION

The AHB to APB Bridge Protocol has been confirmed when connected to external memory, and simulation results are shown along with its functional coverage, estimated to be around 70%. The module modification has no impact on the protocol's operation, and all AHB to APB capabilities have been thoroughly validated. Wrapped bust and incremental burst actions might be enhanced in the future

REFERENCES

[1] Design and Verification of AMBA AHB Peru Malla Giridhar; Priyanka Choudhury 2019 1st International Conference on Advanced Technologies in Intelligent Control, Environment, Computing & Communication Engineering (ICATIECE).

[2] Kiran Rawat, Kanika Sahni and Sujata Pandey, “RTL Implementation for AMBA ASB APB Protocol at System on Chip level”, IEEE International Conference on Signal Processing and Integrated Networks (SPIN), 2015.

[3] Physical Bridge Design for High Performance to Peripheral Bus: An Opensource Approach Arjun Suresh, Somesh Nandi 2020 Third International Conference on Advances in Electronics, Communications (ICAECC) Computers,

[4] P. Jain and S. Rao, "Design and Verification of Advanced Microcontroller Bus Architecture-Advanced Peripheral Bus (AMBA APB) Protocol," 2021 Third International Conference on Intelligent Communication Technologies and Virtual Mobile Networks (ICICV).

[5] Manu B.N and Prabhavathi P, "Design and Implementation of AMBA ASB APB Bridge", IEEE International Conference on Fuzzy Theory and Its Application, pp. 2377-5823, 2013
[6] Karin Avnit, Arcot Sowmya and Jorgen Pedersen, "ACS: Automatic Converter Synthesis for SoC Bus Protocols", Springer-Verlag Berlin Heidelberg, pp. 343– 348, 2010.
[7] Shivani Malhotra and Neelam Rup Prakash, "UVM Based Verification IP of AMBA AXI Protocol Showing Multiple Transactions and Data Interconnect", Springer Nature Singapore Pte Ltd, 201
[8] Vani R. M. and M. Roopa, "Design of AMBA based AHB2APB Bridge", IJCSNS International Journal of Computer Science and Network Security, VOL.10 No.11, November 2010.
[9] Wang Hang Suan, Asral Bahari Jambek, Mohd Nazrin Bin Md Isa, Azizi bin Harun, Shaiful Nizam Bin Mohya and Zulfiqar Ali Bin Abd. Aziz, " Design and implementation of AMBA bridge protocol in System onChip design", Indonesian Journal of Electrical Engineering and Computer Science, Volume 14, No. 2, pp. 788-795, May 2019.
[10] Shivani Malhotra and Neelam Rup Prakash, "UVM Based Verification IP of AMBA AXI Protocol Showing Multiple Transactions and Data Interconnect", Springer Nature Singapore Pte Ltd, 2019.
[11] Aparna Kharade and V. Jayashree, "VLSI Design of AMBA based AHB2APB Bridge", International Journal of VLSI design & Communication Systems (VLSICS) Vol.9, No.3, June 2018.
[12] Zdravko Panjkov, et.al., "OCP2XI Bridge: An OCP to AXI Protocol Bridge", Springer International Publishing Switzerland, pp. 179–190, 2014.
[13] Shankar, Dipti Girdhar and Neeraj Kr. Shukla, "Design and Verification of AMBA APB Protocol", International Journal of Computer Applications, Volume 95, No.21, June 2014.
[14] Karin Avnit, Arcot Sowmya and Jorgen Peddersen, "ACS: Automatic Converter Synthesis for SoC Bus Protocols", Springer-Verlag Berlin Heidelberg, pp. 343–348, 2010.